```
EEEEEEEEEEEEEEE  RRRRRRRRRRR      FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRR      FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  RRRRRRRRRRR      FFFFFFFFFFFFFFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEE              RRR         RRR  FFF
EEEEEEEEEEEE     RRRRRRRRRRR      FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRR      FFFFFFFFFFFF
EEEEEEEEEEEE     RRRRRRRRRRR      FFFFFFFFFFFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR   RRR        FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEE              RRR      RRR     FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
EEEEEEEEEEEEEEE  RRR         RRR  FFF
```

FILEID**TUTAPE

```
TTTTTTTTTT  UU      UU  TTTTTTTTTT    AAAAAA    PPPPPPPP    EEEEEEEEEE
TTTTTTTTTT  UU      UU  TTTTTTTTTT    AAAAAA    PPPPPPPP    EEEEEEEEEE
    TT      UU      UU      TT       AA    AA  PP      PP  EE
    TT      UU      UU      TT       AA    AA  PP      PP  EE
    TT      UU      UU      TT       AA    AA  PP      PP  EE
    TT      UU      UU      TT       AA    AA  PP      PP  EE
    TT      UU      UU      TT       AA    AA  PPPPPPPP    EEEEEEEE
    TT      UU      UU      TT       AA    AA  PPPPPPPP    EEEEEEEE
    TT      UU      UU      TT       AAAAAAAAAA  PP        EE
    TT      UU      UU      TT       AAAAAAAAAA  PP        EE
    TT      UU      UU      TT       AA    AA  PP        EE              ....
    TT      UU      UU      TT       AA    AA  PP        EE              ....
    TT      UUUUUUUUUU      TT       AA    AA  PP        EEEEEEEEEE      ....
    TT      UUUUUUUUUU      TT       AA    AA  PP        EEEEEEEEEE      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

000
000
000
000
000
000
027
027
027
027
027
027
027
027
027
027
028
028
028
028
028
028
028
028
028
028
028
029
029
029
029
029
029
029
029
030
030
030
030
030
030
030
030
030
030
030
031
031
031
031
031
031
031
031
031
031
031
032

```
0001            SUBROUTINE TUTAPE (LUN)
0002    C
0003    C Version:      'V04-000'
0004    C
0005    C*****************************************************************************
0006    C*                                                                           *
0007    C*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0008    C*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0009    C*  ALL RIGHTS RESERVED.                                                     *
0010    C*                                                                           *
0011    C*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0012    C*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0013    C*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0014    C*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0015    C*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0016    C*  TRANSFERRED.                                                             *
0017    C*                                                                           *
0018    C*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0019    C*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0020    C*  CORPORATION.                                                             *
0021    C*                                                                           *
0022    C*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0023    C*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0024    C*                                                                           *
0025    C*                                                                           *
0026    C*****************************************************************************
0027    C
0028    C
0029    C
0030    C       AUTHOR  BRIAN PORTER            CREATION DATE   5-OCT-1979
0031    C
0032    C       Fucntional description:
0033    C
0034    C       This module displays TU58 errors.
0035    C
0036    C       Modified by:
0037    C
0038    C       V03-004 SAR0236         Sharon A. Reynolds,     28-Mar-1984
0039    C               Changed the call to UCB$L_OWNUIC to ORB$L_OWNER.
0040    C
0041    C       V03-003 SAR0130         Sharon A. Reynolds,     8-Sep-1983
0042    C               Changed the overprint carriage control in the 'format'
0043    C               statements for use with ERF.
0044    C
0045    C       V03-002 SAR0100         Sharon A. Reynolds,     20-Jun-1983
0046    C               Changed the carriage control in the 'format' statements
0047    C               for use with ERF.
0048    C
0049    C       V03-001 SAR0052         Sharon A. Reynolds,     13-Jun-1983
0050    C               Removed brief/cryptic support.
0051    C
0052    C       v02-004 BP0004          Brian Porter,           23-NOV-1981
0053    C               Minor edit.
0054    C
0055    C       v02-003 BP0003          Brian Porter,           05-NOV-1981
0056    C               Added 'device attention' support.
0057    C
```

```
0058    C       v02-002 BP0002          Brian Porter,           01-JUL-1981
0059    C               Added call to LOGGER and DHEAD.  Added DIAGNOSTIC_MODE.
0060    C
0061    C       v02-001 BP0001          Brian Porter,           19-JUL-1980
0062    C               Corrected fortran format error when end packet has illegal
0063    C               byte count.  Removed call to UNUSED_BITS.
0064    C**
0065
0066            INCLUDE 'SRC$:MSGHDR.FOR /NOLIST'
0125            INCLUDE 'SRC$:DEVERR.FOR /NOLIST'
0226
0227
0228            BYTE            LUN
0229            BYTE            CMD_PKT(0:13)
0230            BYTE            END_PKT(0:13)
0231
0232            INTEGER*4       FIELD
0233            INTEGER*4       RCSR
0234            INTEGER*4       RBUF
0235            INTEGER*4       XCSR
0236            INTEGER*4       XBUF
0237            INTEGER*4       COMPRESS4
0238            INTEGER*4       COMPRESSC
0239
0240            logical*1       diagnostic_mode
0241
0242            PARAMETER       TIMEOUT = 96
0243            PARAMETER       READ = 2
0244            PARAMETER       WRITE = 3
0245            PARAMETER       POSITION = 5
0246
0247            EQUIVALENCE     (EMB$L_DV_REGSAV(0),XCSR)
0248            EQUIVALENCE     (EMB$L_DV_REGSAV(1),XBUF)
0249            EQUIVALENCE     (EMB$L_DV_REGSAV(2),RCSR)
0250            EQUIVALENCE     (EMB$L_DV_REGSAV(3),RBUF)
0251            EQUIVALENCE     (EMB(98),CMD_PKT(0))
0252            EQUIVALENCE     (EMB(112),END_PKT(0))
0253
0254
0255
0256            CHARACTER*14    UNIT_NUMBER
0257            DATA    UNIT_NUMBER     /'UNIT NUMBER = '/
0258
0259            CHARACTER*11    UNUSED_BYTE
0260            DATA    UNUSED_BYTE     /'UNUSED BYTE'/
0261
0262            CHARACTER*19    CONTROL_PACKET
0263            DATA    CONTROL_PACKET  /'TU58 CONTROL PACKET'/
0264
0265            CHARACTER*14    COMMAND_PACKET
0266            DATA    COMMAND_PACKET  /'COMMAND PACKET'/
0267
0268            CHARACTER*7     OP_CODE
0269            DATA    OP_CODE         /'OP CODE'/
0270
0271            CHARACTER*17    PBYTE_COUNT
0272            DATA    PBYTE_COUNT     /'PACKET BYTE COUNT'/
```

```
0273
0274            CHARACTER*15    SEQUENCE_LO
0275            DATA    SEQUENCE_LO     7'SEQUENCE LO, HI'/
0276
0277            CHARACTER*17    BYTECOUNT_LO
0278            DATA    BYTECOUNT_LO    /'BYTE COUNT LO, HI'/
0279
0280            CHARACTER*19    BLOCK_LO
0281            DATA    BLOCK_LO        /'BLOCK NUMBER LO, HI'/
0282
0283            CHARACTER*15    CHECKSUM_LO
0284            DATA    CHECKSUM_LO     7'CHECKSUM LO, HI'/
0285
0286            CHARACTER*17    V1RCSR(6:7)
0287            DATA    V1RCSR(6)       /'INTERRUPT ENABLE*'/
0288            DATA    V1RCSR(7)       /'RECEIVER DONE*'/
0289
0290            CHARACTER*16    V2RCSR(11:11)
0291            DATA    V2RCSR(11)      /'RECEIVER ACTIVE*'/
0292
0293            CHARACTER*15    V1RBUF(12:15)
0294            DATA    V1RBUF(12)      /'PARITY ERROR*'/
0295            DATA    V1RBUF(13)      /'FRAMING ERROR*'/
0296            DATA    V1RBUF(14)      /'OVER-RUN ERROR*'/
0297            DATA    V1RBUF(15)      /'ERROR*'/
0298
0299            CHARACTER*6     V1XCSR(0:0)
0300            DATA    V1XCSR(0)       /'BREAK*'/
0301
0302            CHARACTER*18    V2XCSR(6:7)
0303            DATA    V2XCSR(6)       /'INTERRUPT ENABLE*'/
0304            DATA    V2XCSR(7)       /'TRANSMITTER READY*'/
0305
0306            CHARACTER*20    V1OP_CODE(-1:12)
0307            DATA    V1OP_CODE(-1)   /'ILLEGAL FUNCTION*'/
0308            DATA    V1OP_CODE(0)    /'NO OPERATION*'/
0309            DATA    V1OP_CODE(1)    /'INITIALIZE*'/
0310            DATA    V1OP_CODE(2)    /'READ*'/
0311            DATA    V1OP_CODE(3)    /'WRITE*'/
0312            DATA    V1OP_CODE(4)    /'COMPARE*'/
0313            DATA    V1OP_CODE(5)    /'POSITION*'/
0314            DATA    V1OP_CODE(6)    /'ABORT*'/
0315            DATA    V1OP_CODE(7)    /'DIAGNOSE*'/
0316            DATA    V1OP_CODE(8)    /'GET STATUS*'/
0317            DATA    V1OP_CODE(9)    /'SET STATUS*'/
0318            DATA    V1OP_CODE(10)   /'GET CHARACTERISTICS*'/
0319            DATA    V1OP_CODE(11)   /'SET CHARACTERISTICS*'/
0320            DATA    V1OP_CODE(12)   /'ILLEGAL FUNCTION*'/
0321
0322            CHARACTER*16    V2OP_CODE(63:65)
0323            DATA    V2OP_CODE(63)   /'ILLEGAL OP CODE*'/
0324            DATA    V2OP_CODE(64)   /'SEND END PACKET*'/
0325            DATA    V2OP_CODE(65)   /'ILLEGAL OP CODE*'/
0326
0327            CHARACTER*10    MODIFIER(0:0)
0328            DATA    MODIFIER(0)     /'DATACHECK*'/
0329
```

```
0330
0331            CHARACTER*18    V1SUM_STATUS(4:7)
0332            DATA    V1SUM_STATUS(4)/'LOGIC ERROR*'/
0333            DATA    V1SUM_STATUS(5)/'MOTION ERROR*'/
0334            DATA    V1SUM_STATUS(6)/'TRANSFER ERROR*'/
0335            DATA    V1SUM_STATUS(7)/'SPECIAL CONDITION*'/
0336
0337            BYTE            V1SUCCESS_CODE(0:11)
0338            DATA    V1SUCCESS_CODE(0)/'00'X/
0339            DATA    V1SUCCESS_CODE(1)/'01'X/
0340            DATA    V1SUCCESS_CODE(2)/'FF'X/
0341            DATA    V1SUCCESS_CODE(3)/'FE'X/
0342            DATA    V1SUCCESS_CODE(4)/'F8'X/
0343            DATA    V1SUCCESS_CODE(5)/'F7'X/
0344            DATA    V1SUCCESS_CODE(6)/'F5'X/
0345            DATA    V1SUCCESS_CODE(7)/'EF'X/
0346            DATA    V1SUCCESS_CODE(8)/'E0'X/
0347            DATA    V1SUCCESS_CODE(9)/'DF'X/
0348            DATA    V1SUCCESS_CODE(10)/'D0'X/
0349            DATA    V1SUCCESS_CODE(11)/'C9'X/
0350
0351            CHARACTER*22    V2SUCCESS_CODE(0:12)
0352            DATA    V2SUCCESS_CODE(0)/'NORMAL COMPLETION*'/
0353            DATA    V2SUCCESS_CODE(1)/'RETRY COMPLETION*'/
0354            DATA    V2SUCCESS_CODE(2)/'SELF TEST FAILURE*'/
0355            DATA    V2SUCCESS_CODE(3)/'PARTIAL OPERATION*'/
0356            DATA    V2SUCCESS_CODE(4)/'INVALID UNIT NUMBER*'/
0357            DATA    V2SUCCESS_CODE(5)/'NO CARTRIDGE PRESENT*'/
0358            DATA    V2SUCCESS_CODE(6)/'WRITE PROTECTED*'/
0359            DATA    V2SUCCESS_CODE(7)/'DATA CHECK*'/
0360            DATA    V2SUCCESS_CODE(8)/'BLOCK NOT FOUND*'/
0361            DATA    V2SUCCESS_CODE(9)/'MOTOR STOPPED*'/
0362            DATA    V2SUCCESS_CODE(10)/'INVALID OP CODE*'/
0363            DATA    V2SUCCESS_CODE(11)/'INVALID RECORD NUMBER*'/
0364            DATA    V2SUCCESS_CODE(12)/'ILLEGAL SUCCESS CODE*'/
0365
0366            CHARACTER*9     BYTE_COUNT(9:11)
0367            DATA    BYTE_COUNT(9)   /' ILLEGAL*'/
0368            DATA    BYTE_COUNT(10)  /' = 10.*'/
0369            DATA    BYTE_COUNT(11)  /' ILLEGAL*'/
0370
0371
0372            CALL FRCTOF (LUN)
0373
0374            if (emb$t_dv_name(1:3) .eq. 'CSA') then
0375
0376            call dhead1 (lun,'CONSOLE TU58')
0377            else
0378
0379            call dhead1 (lun,'UBA TU58')
0380            endif
0381
0382            diagnostic_mode = .false.
0383
0384            if (lib$extzv(2,1,xcsr) .eq. 1) diagnostic_mode = .true.
0385
0386            CALL LINCHK (LUN,2)
```

```
0387
0388            WRITE(LUN,10) RCSR
0389    10      FORMAT(/' ',T8,'RCSR',T24,Z8.4)
0390
0391            if (.not. diagnostic_mode) then
0392
0393            CALL OUTPUT (LUN,RCSR,V1RCSR,6,6,7,'0')
0394
0395            CALL OUTPUT (LUN,RCSR,V2RCSR,11,11,11,'0')
0396            endif
0397
0398            CALL LINCHK (LUN,1)
0399
0400            WRITE(LUN,15) RBUF
0401    15      FORMAT(' ',T8,'RBUF',T24,Z8.4)
0402
0403            if (.not. diagnostic_mode) then
0404
0405            DO 19,I = 12,15
0406
0407            IF (JIAND(RBUF,2**I) .NE. 0) THEN
0408
0409            CALL LINCHK (LUN,1)
0410
0411            WRITE(LUN,18) V1RBUF(I)
0412    18      FORMAT(' ',T40,'RECEIVE DATA ',A<COMPRESSC (V1RBUF(I))>)
0413            ENDIF
0414
0415    19      CONTINUE
0416            endif
0417
0418            CALL LINCHK (LUN,1)
0419
0420            WRITE(LUN,20) XCSR
0421    20      FORMAT(' ',T8,'XCSR',T24,Z8.4)
0422
0423            if (.not. diagnostic_mode) then
0424
0425            CALL OUTPUT (LUN,XCSR,V1XCSR,0,0,0,'0')
0426
0427            CALL OUTPUT (LUN,XCSR,V2XCSR,6,6,7,'0')
0428            else
0429
0430            CALL LINCHK (LUN,1)
0431
0432            WRITE(LUN,25)
0433    25      FORMAT(' ',T40,'DIAGNOSTIC MODE')
0434            endif
0435
0436            CALL LINCHK (LUN,1)
0437
0438            WRITE (LUN,30) XBUF
0439    30      FORMAT(' ',T8,'XBUF',T24,Z8.4)
0440
0441            if (.not. diagnostic_mode) then
0442
0443            CALL LINCHK (LUN,3)
```

```
0444
0445            IF (CMD_PKT(0) .NE. 2) THEN
0446
0447            WRITE(LUN,40)
0448    40      FORMAT(/' ','COMMAND PACKET HAS INVALID FLAG',/)
0449
0450            DO 44,I = 0,14
0451
0452            CALL LINCHK (LUN,1)
0453
0454            WRITE(LUN,42) CMD_PKT(I)
0455    42      FORMAT(' ',T30,Z2.2)
0456
0457    44      CONTINUE
0458            ELSE
0459
0460            WRITE(LUN,45)
0461    45      FORMAT(/' ','COMMAND MESSAGE PACKET',/)
0462
0463            CALL LINCHK (LUN,2)
0464
0465            WRITE(LUN,50) CMD_PKT(0),CONTROL_PACKET
0466    50      FORMAT(' ',T8,'FLAG',T30,Z2.2,/,T40,A19)
0467
0468            CALL LINCHK (LUN,2)
0469
0470            FIELD = LIB$EXTZV(0,8,CMD_PKT(1))
0471
0472            WRITE(LUN,55) PBYTE_COUNT,CMD_PKT(1),PBYTE_COUNT,
0473           1 BYTE_COUNT(MAX(9,MIN(11,FIELD)))
0474    55      FORMAT(' ',T8,A17,T30,Z2.2,/,
0475           1 T40,A17,A<COMPRESSC (BYTE_COUNT(MAX(9,MIN(11,FIELD))))>)
0476
0477            CALL LINCHK (LUN,2)
0478
0479            FIELD = LIB$EXTZV(0,8,CMD_PKT(2))
0480
0481            WRITE(LUN,60) OP_CODE,CMD_PKT(2),V1OP_CODE(MAX(-1,MIN(11,FIELD)))
0482    60      FORMAT(' ',T8,A7,T30,Z2.2,/,T40,
0483           1 'FUNCTION = ',A<COMPRESSC (V1OP_CODE(MAX(-1,MIN(11,FIELD))))>)
0484
0485            CALL LINCHK (LUN,1)
0486
0487            WRITE(LUN,65) CMD_PKT(3)
0488    65      FORMAT(' ',T8,'OP CODE MODIFIER',T30,Z2.2)
0489
0490            CALL OUTPUT (LUN,CMD_PKT(3),MODIFIER,0,0,0,'0')
0491
0492            CALL LINCHK (LUN,2)
0493
0494            FIELD = LIB$EXTZV(0,8,CMD_PKT(4))
0495
0496            WRITE(LUN,68) CMD_PKT(4),UNIT_NUMBER,FIELD
0497    68      FORMAT(' ',T8,'UNIT',T30,Z2.2,/,
0498           1 T40,A14,I<COMPRESS4 (FIELD)>,'.')
0499
0500            CALL LINCHK (LUN,1)
```

```
0501
0502            WRITE(LUN,70) UNUSED_BYTE,CMD_PKT(5)
0503     70     FORMAT(' ',T8,A11,T30,Z2.2)
0504
0505            CALL LINCHK (LUN,2)
0506
0507            WRITE(LUN,75) SEQUENCE_LO,CMD_PKT(6),CMD_PKT(7)
0508     75     FORMAT(' ',T8,A15,T30,Z2.2,/,T30,Z2.2)
0509
0510            CALL LINCHK (LUN,2)
0511
0512            WRITE(LUN,80) BYTECOUNT_LO,CMD_PKT(8)
0513     80     FORMAT(' ',T8,A17,T30,Z2.2)
0514
0515            Field = ' '
0516
0517            IF (CMD_PKT(2) .EQ. READ
0518            1 .OR.
0519            2 CMD_PKT(2) .EQ. WRITE) THEN
0520
0521            FIELD = LIB$EXTZV(0,16,CMD_PKT(8))
0522            ENDIF
0523
0524            WRITE(LUN,85) CMD_PKT(9),FIELD
0525     85     FORMAT(' ',T30,Z2.2,:,T40,
0526            1 'TRANSFER BYTE COUNT = ',I<COMPRESS4 (FIELD)>,'.')
0527
0528            CALL LINCHK (LUN,2)
0529
0530            WRITE(LUN,90) BLOCK_LO,CMD_PKT(10)
0531     90     FORMAT(' ',T8,A19,T30,Z2.2)
0532
0533            Field = ' '
0534
0535            IF (CMD_PKT(2) .EQ. POSITION
0536            1 .OR.
0537            2 CMD_PKT(2) .EQ. READ
0538            3 .OR.
0539            4 CMD_PKT(2) .EQ. WRITE) THEN
0540
0541            FIELD = LIB$EXTZV(0,16,CMD_PKT(10))
0542            ENDIF
0543
0544            WRITE(LUN,95) CMD_PKT(11),FIELD
0545     95     FORMAT(' ',T30,Z2.2,:,T40,
0546            1 'REQUESTED BLOCK = ',I<COMPRESS4 (FIELD)>,'.')
0547
0548            CALL LINCHK (LUN,2)
0549
0550            WRITE(LUN,100) CHECKSUM_LO,CMD_PKT(12),CMD_PKT(13)
0551     100    FORMAT(' ',T8,A15,T30,Z2.2,/,T30,Z2.2)
0552            ENDIF
0553
0554
0555            CALL LINCHK (LUN,3)
0556
0557            IF (END_PKT(0) .NE. 2) THEN
```

```
0558
0559            WRITE(LUN,105)
0560      105   FORMAT(/' ','END PACKET HAS INVALID FLAG',/)
0561
0562            DO 115,I = 0,13
0563
0564            CALL LINCHK (LUN,1)
0565
0566            WRITE(LUN,110) END_PKT(I)
0567      110   FORMAT(' ',T30,Z2.2)
0568
0569      115   CONTINUE
0570            ELSE
0571
0572            WRITE(LUN,120)
0573      120   FORMAT(/' ','END MESSAGE PACKET',/)
0574
0575            CALL LINCHK (LUN,2)
0576
0577            WRITE(LUN,125) END_PKT(0),CONTROL_PACKET
0578      125   FORMAT(' ',T8,'FLAG',T30,Z2.2,/,T40,A19)
0579
0580            CALL LINCHK (LUN,2)
0581
0582            FIELD = LIB$EXTZV(0,8,END_PKT(1))
0583
0584            WRITE(LUN,130) PBYTE_COUNT,END_PKT(1),PBYTE_COUNT,
0585           1 BYTE_COUNT(MAX(9,MIN(11,FIELD)))
0586      130   FORMAT(' ',T8,A17,T30,Z2.2,/,
0587           1 T40,A17,A<COMPRESSC (BYTE_COUNT(MAX(9,MIN(11,FIELD))))>)
0588
0589            CALL LINCHK (LUN,2)
0590
0591            FIELD = LIB$EXTZV(0,8,END_PKT(2))
0592
0593            WRITE(LUN,135) OP_CODE,END_PKT(2),V2OP_CODE(MAX(63,MIN(65,FIELD)))
0594      135   FORMAT(' ',T8,A7,T30,Z2.2,7,
0595           1 T40,A<COMPRESSC (V2OP_CODE(MAX(63,MIN(65,FIELD))))>)
0596
0597            CALL LINCHK (LUN,2)
0598
0599            WRITE(LUN,140) END_PKT(3)
0600      140   FORMAT(' ',T8,'SUCCESS CODE',T30,Z2.2)
0601
0602            J = 12
0603
0604            DO 155,I = 0,11
0605
0606            IF (END_PKT(3) .EQ. V1SUCCESS_CODE(I)) J = I
0607
0608      155   CONTINUE
0609
0610            WRITE(LUN,160) V2SUCCESS_CODE(J)
0611      160   FORMAT(' ',T40,A<COMPRESSC (V2SUCCESS_CODE(J))>)
0612
0613            CALL LINCHK (LUN,2)
0614
```

```
0615            FIELD = LIB$EXTZV(0,8,END_PKT(4))
0616
0617            WRITE(LUN,170) END_PKT(4),UNIT_NUMBER,FIELD
0618      170   FORMAT(' ',T8,'UNIT',T30,Z2.2,7,T40,A14,
0619            1 I<COMPRESS4 (FIELD)>,'.')
0620
0621            CALL LINCHK (LUN,1)
0622
0623            WRITE(LUN,175) UNUSED_BYTE,END_PKT(5)
0624      175   FORMAT(' ',T8,A11,T30,Z2.2)
0625
0626            CALL LINCHK (LUN,2)
0627
0628            WRITE(LUN,180) SEQUENCE_LO,END_PKT(6),END_PKT(7)
0629      180   FORMAT(' ',T8,A15,T30,Z2.2,/,T30,Z2.2)
0630
0631            CALL LINCHK (LUN,2)
0632
0633            FIELD = LIB$EXTZV(0,16,END_PKT(8))
0634
0635            WRITE(LUN,185) BYTECOUNT_LO,END_PKT(8),END_PKT(9),FIELD
0636      185   FORMAT(' ',T8,A17,T30,Z2.2,/,
0637            1 T30,Z2.2,T40,'BYTES TRANSFERED = ',I<COMPRESS4 (FIELD)>,'.')
0638
0639            CALL LINCHK (LUN,2)
0640
0641            WRITE(LUN,190) END_PKT(10),END_PKT(11)
0642      190   FORMAT(' ',T8,'STATUS',T30,Z2.2,/,T30,Z2.2)
0643
0644            CALL OUTPUT (LUN,END_PKT(12),V1SUM_STATUS,4,4,7,'0')
0645
0646            CALL LINCHK (LUN,2)
0647
0648            WRITE(LUN,195) CHECKSUM_LO,END_PKT(12),END_PKT(13)
0649      195   FORMAT(' ',T8,A15,T30,Z2.2,/,T30,Z2.2)
0650            ENDIF
0651            endif
0652
0653            call linchk (lun,1)
0654
0655            write(lun,200)
0656      200   format(' ',:)
0657
0658            if (emb$w_hd_entry .ne. 98) then
0659
0660            call ucb$b_ertcnt (lun,emb$b_dv_ertcnt)
0661
0662            call ucb$b_ertmax (lun,emb$b_dv_ertmax)
0663            endif
0664
0665            call orb$l_owner (lun,emb$l_dv_ownuic)
0666
0667            call ucb$l_char (lun,emb$l_dv_char)
0668
0669            call ucb$w_sts (lun,emb$w_dv_sts)
0670
0671            call ucb$l_opcnt (lun,emb$l_dv_opcnt)
```

```
0672
0673            call ucb$w_errcnt (lun,emb$w_dv_errcnt)
0674
0675            if (emb$w_hd_entry .ne. 98) then
0676
0677            call ucb$l_media (lun,emb$l_dv_media)
0678
0679            call linchk (lun,1)
0680
0681            write(lun,200)
0682
0683            call tutape_qio (lun,emb$w_dv_func)
0684
0685            call irp$w_bcnt (lun,emb$w_dv_bcnt)
0686
0687            call irp$w_boff (lun,emb$w_dv_boff)
0688
0689            call irp$l_pid (lun,emb$l_dv_rqpid)
0690
0691            call irp$q_iosb (lun,emb$l_dv_iosb1)
0692            endif
0693
0694            RETURN
0695            END
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	3247	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	906	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	2008	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	EMB	512	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	6673	

ENTRY POINTS

Address	Type	Name
0-00000000		TUTAPE

VARIABLES

Address	Type	Name	Address	Type	Name
2-000003EA	CHAR	BLOCK_LO	2-000003D9	CHAR	BYTECOUNT_LO
2-000003FD	CHAR	CHECKSUM_LO	2-000003A4	CHAR	COMMAND_PACKET
2-00000391	CHAR	CONTROL_PACKET	2-00000377	L*1	DIAGNOSTIC_MODE
3-0000001C	L*1	EMB$B_DV_CLASS	3-00000010	L*1	EMB$B_DV_ERTCNT
3-00000011	L*1	EMB$B_DV_ERTMAX	3-0000003E	L*1	EMB$B_DV_NAMLNG
3-0000003A	L*1	EMB$B_DV_SLAVE	3-0000001D	L*1	EMB$B_DV_TYPE

```
  3-00000036  I*4   EMB$L_DV_CHAR
  3-00000016  I*4   EMB$L_DV_IOSB2
  3-0000004E  I*4   EMB$L_DV_NUMREG
  3-00000032  I*4   EMB$L_DV_OWNUIC
  3-00000000  I*4   EMB$L_HD_SID
  3-00000024  I*2   EMB$W_DV_BCNT
  3-0000002C  I*2   EMB$W_DV_ERRCNT
  3-0000001A  I*2   EMB$W_DV_STS
  3-00000004  I*2   EMB$W_HD_ENTRY
  2-0000040C  I*4   FIELD
  2-00000414  I*4   J
  2-000003B2  CHAR  OP_CODE
  3-0000005E  I*4   RBUF
  2-000003CA  CHAR  SEQUENCE_LO
  2-00000386  CHAR  UNUSED_BYTE
  3-00000052  I*4   XCSR
  3-00000012  I*4   EMB$L_DV_IOSB1
  3-00000026  I*4   EMB$L_DV_MEDIA
  3-0000002E  I*4   EMB$L_DV_OPCNT
  3-0000001E  I*4   EMB$L_DV_RQPID
  3-0000003F  CHAR  EMB$T_DV_NAME
  3-00000022  I*2   EMB$W_DV_BOFF
  3-0000003C  I*2   EMB$W_DV_FUNC
  3-0000002A  I*2   EMB$W_DV_UNIT
  3-0000000E  I*2   EMB$W_HD_ERRSEQ
  2-00000410  I*4   I
AP-00000004@  L*1   LUN
  2-000003B9  CHAR  PBYTE_COUNT
  3-0000005A  I*4   RCSR
  2-00000378  CHAR  UNIT_NUMBER
  3-00000056  I*4   XBUF
```

ARRAYS

```
    Address  Type  Name                    Bytes  Dimensions
  2-0000035C  CHAR  BYTE_COUNT                27  (9:11)
  3-00000062  L*1   CMD_PKT                   14  (0:13)
  3-00000000  L*1   EMB                      512  (0:511)
  3-00000052  I*4   EMB$L_DV_REGSAV          420  (0:104)
  3-00000006  I*4   EMB$Q_HD_TIME              8  (2)
  3-00000070  L*1   END_PRT                   14  (0:13)
  2-000001E0  CHAR  MODIFIER                  10  (0:0)
  2-00000098  CHAR  V1OP_CODE                280  (-1:12)
  2-00000032  CHAR  V1RBUF                    60  (12:15)
  2-00000000  CHAR  V1RCSR                    34  (6:7)
  2-00000232  L*1   V1SUCCESS_CODE            12  (0:11)
  2-000001EA  CHAR  V1SUM_STATUS              72  (4:7)
  2-0000006E  CHAR  V1XCSR                     6  (0:0)
  2-000001B0  CHAR  V2OP_CODE                 48  (63:65)
  2-00000022  CHAR  V2RCSR                    16  (11:11)
  2-0000023E  CHAR  V2SUCCESS_CODE           286  (0:12)
  2-00000074  CHAR  V2XCSR                    36  (6:7)
```

LABELS

```
    Address  Label     Address  Label     Address  Label     Address  Label     Address  Label     Address  Label
  1-00000040  10'    1-00000052  15'    1-00000063  18'       **    19     1-0000007E  20'    1-0000008F  25'
  1-000000A6  30'    1-000000B7  40'    1-000000DE  42'       **    44     1-000000E7  45'    1-00000105  50'
  1-0000011B  55'    1-00000133  60'    1-00000156  65'    1-00000173  68'    1-00000192  70'    1-0000019F  75'
  1-000001B2  80'    1-000001BF  85'    1-000001EC  90'    1-000001F9  95'    1-00000222  100'   1-00000235  105'
  1-00000258  110'      **       115'   1-00000261  120'   1-0000027B  125'   1-00000291  130'   1-000002A9  135'
  1-000002BF  140'      **       155'   1-000002D8  160'   1-000002E4  170'   1-00000303  175'   1-00000310  180'
  1-00000323  185'   1-00000356  190'   1-0000036F  195'   1-00000382  200'
```

FUNCTIONS AND SUBROUTINES REFERENCED

Type	Name	Type	Name	Type	Name	Type	Name	Type	Name	Type	Name
I*4	COMPRESS4	I*4	COMPRESSC		DHEAD1		FRCTOF		IRP$L_PID		IRP$Q_IOSB
	IRP$W_BCNT		IRP$W_BOFF	I*4	LIB$EXTZV		LINCHK		ORB$L_OWNER		OUTPUT
	TUTAPE_QIO		UCB$B_ERTCNT		UCB$B_ERTMAX		UCB$L_CHAR		UCB$L_MEDIA		UCB$L_OPCNT
	UCB$W_ERRCNT		UCB$W_STS								

```
0001
0002
0003
0004          Subroutine TUTAPE_QIO (lun,emb$w_dv_func)
0005          include 'src$:qiocommon.for /nolist'
0270          byte            lun
0271          integer*2       emb$w_dv_func
0272          integer*4       qiocode(0:1,0:63)
0273
0274
0275          if (qiocode(0,0) .eq. 0) then
0276          qiocode(1,09) = %loc(io$_search)
0277          qiocode(1,11) = %loc(io$_writepblk)
0278          qiocode(1,12) = %loc(io$_readpblk)
0279          qiocode(1,26) = %loc(io$_setchar)
0280          qiocode(1,27) = %loc(io$_sensechar)
0281          qiocode(1,29) = %loc(io$_diagnose)
0282          qiocode(1,32) = %loc(io$_writelblk)
0283          qiocode(1,33) = %loc(io$_readlblk)
0284          qiocode(1,35) = %loc(io$_setmode)
0285          qiocode(1,39) = %loc(io$_sensemode)
0286          qiocode(1,48) = %loc(io$_writevblk)
0287          qiocode(1,49) = %loc(io$_readvblk)
0288          qiocode(1,50) = %loc(io$_access)
0289          qiocode(1,51) = %loc(io$_create)
0290          qiocode(1,52) = %loc(io$_deaccess)
0291          qiocode(1,53) = %loc(io$_delete)
0292          qiocode(1,54) = %loc(io$_modify)
0293          qiocode(1,56) = %loc(io$_acpcontrol)
0294          qiocode(1,57) = %loc(io$_mount)
0295          do 10,i = 0,63
0296
0297
0298
0299
0300
0301
0302
0303
0304
0305
0306
0307
0308
0309
0310
0311
0312
0313
0314
0315
0316
0317
0318
0319
0320
```

```
0321
0322            qiocode(0,i) = 33
0323
0324            if (qiocode(1,i) .eq. 0) then
0325
0326            qiocode(1,i) = %loc(qio_string)
0327            endif
0328
0329     10     continue
0330            endif
0331
0332            call irp$w_func (lun,emb$w_dv_func,
0333            1 qiocode(0,lib$extzv(0,6,emb$w_dv_func)))
0334
0335            return
0336
0337            end
```

PROGRAM SECTIONS

	Name	Bytes	Attributes
0	$CODE	225	PIC CON REL LCL SHR EXE RD NOWRT LONG
1	$PDATA	8	PIC CON REL LCL SHR NOEXE RD NOWRT LONG
2	$LOCAL	548	PIC CON REL LCL NOSHR NOEXE RD WRT LONG
3	QIOCOMMON	1247	PIC OVR REL GBL SHR NOEXE RD WRT LONG
	Total Space Allocated	2028	

ENTRY POINTS

Address	Type	Name
0-00000000		TUTAPE_QIO

VARIABLES

Address	Type	Name	Address	Type	Name
AP-0000008a	I*2	EMB$W_DV_FUNC	2-00000200	I*4	I
3-00000442	CHAR	IO$_ABORT	3-0000034D	CHAR	IO$_ACCESS
3-000003C2	CHAR	IO$_ACPCONTROL	3-000004B3	CHAR	IO$_AVAILABLE
3-00000297	CHAR	IO$_CLEAN	3-00000369	CHAR	IO$_CREATE
3-00000385	CHAR	IO$_DEACCESS	3-00000393	CHAR	IO$_DELETE
3-0000026D	CHAR	IO$_DIAGNOSE	3-00000065	CHAR	IO$_DRVCLR
3-000004CB	CHAR	IO$_DSE	3-000000A9	CHAR	IO$_ERASETAPE
3-00000276	CHAR	IO$_FORMAT	3-00000071	CHAR	IO$_INITIALIZE
3-00000014	CHAR	IO$_LOADMCODE	3-000003A1	CHAR	IO$_MODIFY
3-000003E2	CHAR	IO$_MOUNT	3-00000000	CHAR	IO$_NOP
3-0000009D	CHAR	IO$_OFFSET	3-000000EB	CHAR	IO$_PACKACK
3-000000E0	CHAR	IO$_QSTOP	3-000003EF	CHAR	IO$_RDSTATS
3-00000421	CHAR	IO$_READCSR	3-00000169	CHAR	IO$_READHEAD

```
  3-000002B6  CHAR IO$_READLBLK
  3-00000200  CHAR IO$_READPRESET
  3-0000033A  CHAR IO$_READVBLK
  3-00000484  CHAR IO$_READWTHXBUF
  3-0000007C  CHAR IO$_RELEASE
  3-000001B8  CHAR IO$_REREADP
  3-000002E6  CHAR IO$_REWIND
  3-000000FC  CHAR IO$_SEARCH
  3-00000231  CHAR IO$_SENSECHAR
  3-0000021D  CHAR IO$_SETCHAR
  3-00000088  CHAR IO$_SETCLOCKP
  3-000002ED  CHAR IO$_SKIPFILE
  3-00000029  CHAR IO$_SPACEFILE
  3-000003D7  CHAR IO$_STARTDATA
  3-00000037  CHAR IO$_STARTMPROC
  3-00000059  CHAR IO$_STOP
  3-0000046B  CHAR IO$_WRITEBUFNCRC
  3-000001E4  CHAR IO$_WRITECHECKH
  3-00000153  CHAR IO$_WRITEHEAD
  3-00000247  CHAR IO$_WRITEMARK
  3-0000012A  CHAR IO$_WRITEPBLK
  3-0000017E  CHAR IO$_WRITETRACKD
  3-00000448  CHAR IO$_WRITEWTHBUF
 AP-00000004@ L*1  LUN
  3-0000013F  CHAR IO$_READPBLK
  3-00000195  CHAR IO$_READTRACKD
  3-0000045A  CHAR IO$_READWTHBUF
  3-0000004D  CHAR IO$_RECAL
  3-000001AB  CHAR IO$_REREADN
  3-000000CA  CHAR IO$_RETCENTER
  3-000002C9  CHAR IO$_REWINDOFF
  3-00000024  CHAR IO$_SEEK
  3-00000309  CHAR IO$_SENSEMODE
  3-000003B8  CHAR IO$_SETCLOCK
  3-000002DD  CHAR IO$_SETMODE
  3-000002FA  CHAR IO$_SKIPRECORD
  3-0000010E  CHAR IO$_SPACERECORD
  3-000000B4  CHAR IO$_STARTDATAP
  3-0000020F  CHAR IO$_STARTSPNDL
  3-0000000D  CHAR IO$_UNLOAD
  3-0000011E  CHAR IO$_WRITECHECK
  3-000003FF  CHAR IO$_WRITECSR
  3-000002A2  CHAR IO$_WRITELBLK
  3-00000314  CHAR IO$_WRITEOF
  3-000001C9  CHAR IO$_WRITERET
  3-00000326  CHAR IO$_WRITEVBLK
  3-00000257  CHAR IO$_WRTTMKR
  3-000004A1  CHAR QIO_STRING

ARRAYS

    Address  Type  Name                 Bytes  Dimensions

  2-00000000  I*4  QIOCODE                512  (0:1, 0:63)

LABELS

    Address   Label

      **      10

FUNCTIONS AND SUBROUTINES REFERENCED

  Type  Name              Type  Name

        IRP$W_FUNC         I*4  LIB$EXTZV

COMMAND QUALIFIERS

  FORTRAN /LIS=LIS$:TUTAPE/OBJ=OBJ$:TUTAPE MSRC$:TUTAPE

  /CHECK=(NOBOUNDS,OVERFLOW,NOUNDERFLOW)
  /DEBUG=(NOSYMBOLS,TRACEBACK)
  /STANDARD=(NOSYNTAX,NOSOURCE_FORM)
  /SHOW=(NOPREPROCESSOR,NOINCLUDE,MAP)
  /F77  /NOG_FLOATING  /I4  /OPTIMIZE  /WARNINGS  /NOD_LINES  /NOCROSS_REFERENCE  /NOMACHINE_CODE  /CONTINUATIONS=19
```

COMPILATION STATISTICS

Run Time:	11.14 seconds
Elapsed Time:	29.09 seconds
Page Faults:	266
Dynamic Memory:	251 pages

0154 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TIMCMP
LIS

TUTAPE
LIS

SBI
LIS

STSEVENT
LIS

TRNS_BITS
LIS

UBAERR
LIS

TIMRB
LIS

UBA
LIS

TU81SENSE
LIS

UNDEFINED
LIS

SYSPWRFL
LIS

SUMMARY
LIS

UNKN_DISP
LIS

SHRVECTOR
LIS

SYSTARTUP
LIS

UBAINT
LIS

TOF
LIS

UNKNOWN
LIS

TSTAPE
LIS